PART B REPORT

Name: Anant Gupta **Student ID:** 1824943

1. Methods

1.1 Dataset and Splits

The full CIFAR-10 dataset (50 000 training, 10 000 test images) was used.
A 45 000 / 5 000 train-validation split was created using `torch.utils.data.random_split`.
All images were normalized using CIFAR-10 channel means and standard deviations.

1.2 Data Augmentations

Three augmentations were applied to the training set:

- **RandomCrop(32, padding=4)** - Adds translation invariance. Randomly shifts the image by up to 4 pixels in any direction. Helps model learn features regardless of position
- **RandomHorizontalFlip(p = 0.5)** – Adds horizontal reflection invariance, 50% chance to flip image left-right. Useful for CIFAR-10 (airplane, car, ship look similar flipped)
- **ColorJitter(brightness = 0.2, contrast = 0.2, saturation = 0.2)** – Adds lighting/color robustness. Randomly adjusts brightness, contrast, and saturation. Helps model handle different lighting conditions

All of the three augmentations were disabled for validation and test sets.

1.3 Model Architecture

Class: CIFAR10_CNN_Improved

```
Conv(3→64, kernel_size=3, padding=1) → BN → ReLU → MaxPool(2)
Conv(64→128, kernel_size=3, padding=1) → BN → ReLU → MaxPool(2)
Conv(128→256, kernel_size=3, padding=1) → BN → ReLU
Conv(256→256, kernel_size=3, padding=1) → BN → ReLU → GlobalAvgPool
Dropout(0.5)
Linear(256→10)
```

All convolutions use stride = 1. Global average pooling reduces spatial dimensions to 1×1 before the final fully-connected layer.

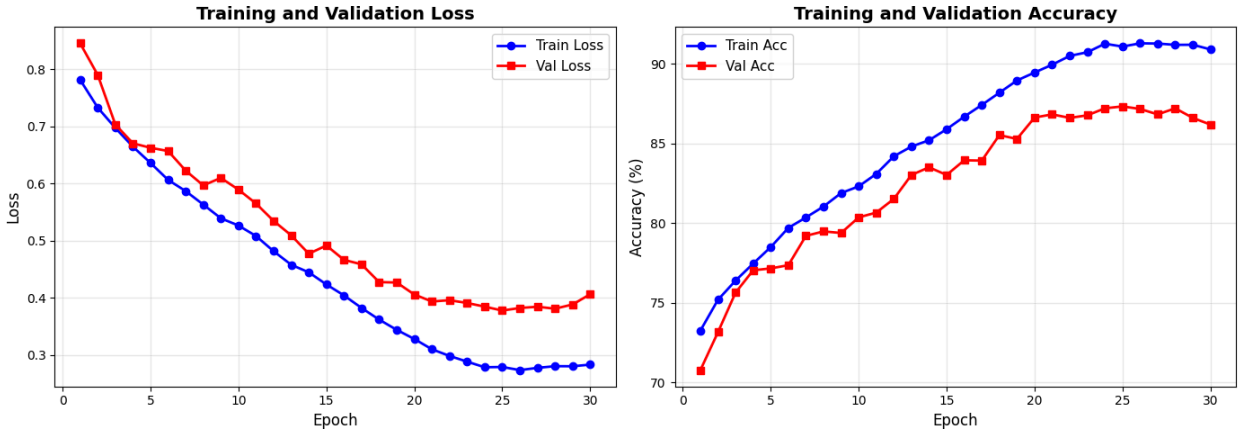
1.3 Hyperparameters

Hyperparameters:
- Epochs: 30
- Batch size: 256
- Learning rate: 0.001 (with Cosine Annealing)
- Weight decay: 0.0005
- Dropout: 0.5

---

2. Results

2.1 Training Curves



Training converged smoothly with BatchNorm and Dropout preventing overfitting.
Validation accuracy plateaued around **≈ 87 %** after epoch 25.

2.2 Final Metrics

Data Splits

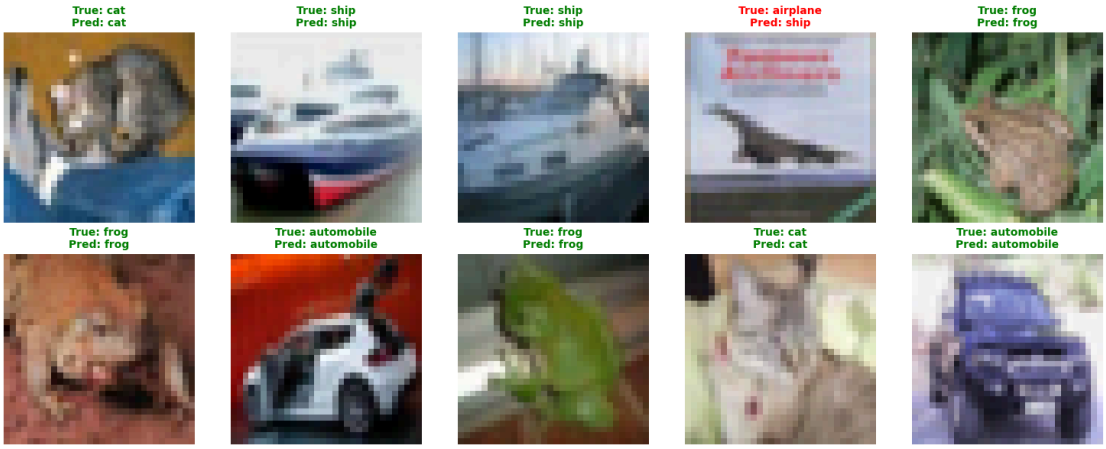
- Training set: 45,000 images (CIFAR-10 train minus 5,000 held-out for validation)
- Validation set: 5,000 images (from original training set)
- Test set: 10,000 images (CIFAR-10 test set)

Final Metrics

```
Training time: 15.27 minutes (0.25 hours)
Best epoch: 25/30
Best validation accuracy: 87.32%
Final test accuracy: 86.88%
```

2.3 Example Predictions

A 3×3 grid of example predictions with correct/incorrect cases.



---

3. Comparison to Assignment 2 (FCNN)

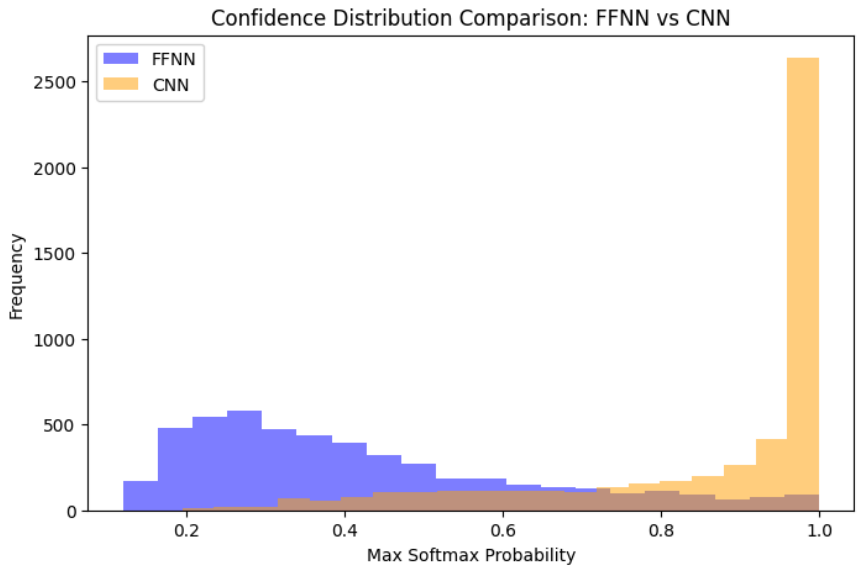
Performance Summary

Model	Train Acc	Val Acc	Train Loss	Val Loss
FFNN	~44%	~47%	~1.6	~1.5
CNN	~90%	~86%	~0.32	~0.4

Analysis

The fully connected network achieves a final validation accuracy of 46.82%, slightly lower than its training accuracy of 43.48%, indicating mild overfitting. The model's predictions also tend to have lower confidence, as expected for a network that ignores spatial structure in images. Flattening the input removes all spatial relationships, which limits the network's ability to recognize local patterns like edges or textures—critical features for image classification.

CNNs outperform fully connected networks in this setting because they exploit **spatial bias** through convolutions, **parameter efficiency** via weight sharing, and can leverage **data augmentation** to improve generalization. Convolutional layers capture local hierarchies, making them better suited for images, while fully connected layers require far more parameters to model the same complexity, increasing overfitting risk. As a result, CNNs achieve higher accuracy and better-calibrated predictions on the same dataset.



---

4. Ablation Study

Removing **Batch Normalization** reduced validation accuracy by ≈ 5–7 % and made training oscillatory, confirming its stabilizing role.
Disabling **augmentations** caused overfitting: training accuracy rose but test accuracy dropped by ≈ 3–4 %.
Lowering **Dropout** from 0.5 to 0.2 improved training speed but reduced test accuracy slightly.
Hence, BatchNorm + Dropout + Augmentations are critical for achieving robust generalization on CIFAR-10.

---

5. Reproducibility Notes

- **Random Seed:** 1
- **Environment:** Google Colab (Torch 2.x, CUDA 11)
- **Data:** CIFAR-10 via `torchvision.datasets`

Note: Used Claude.ai — assisted with boilerplate and debugging, code explanations, analysis guidance, and report structuring.